CLAIMS

What is claimed is:

1. A method for dynamically modifying a mark-up language document during runtime comprising the steps of:

a) providing an insertion tag for use in the mark-up language document; wherein the insertion tag specifies a location in the mark-up language document for inserting data that is unavailable when the mark-up language document is created;

b) receiving a mark-up language document that includes at least one set of insertion tags;

c) receiving data during run-time; and

d) dynamically modifying at run-time the mark-up language document with the received data.

2. The method of claim 1 wherein the step of dynamically modifying at run-time the mark-up language document with the received data includes the step of injecting the received data in the mark-up language document at the location specified by the insertion tags.

3. The method of claim 1 further comprising the step of:

e) using the received data in another portion of the mark-up language document.

4. The method of claim 1 wherein the mark-up language document is an XML document.

5. The method of claim 4 wherein the XML document is an XML test suite file.

6. The method of claim 1 wherein the step of dynamically modifying at run-time the mark-up language document with the received data includes the steps of

parsing the mark-up language document to generate a representation thereof; and

modifying the representation with the received data.

7. The method of claim 6 wherein the step of modifying the representation with the received data includes the steps of

creating a new node that includes the received data;

adding the new node to the representation; and

removing the insertion tag from the representation.

8. A method for dynamically modifying a mark-up language document during runtime comprising the steps of:

receiving a mark-up language document that includes at least one injection tag for specifying a location in the mark-up language document to inject information that is not known at the time that the mark-up language document is created;

generating a representation of the mark-up language document that includes a tree structure;

receiving information; and

injecting the information at the location specified by the injection tag.

9. The method of claim 8 wherein the step of injecting the information at the location specified by the injection tag includes

creating an actual response node;

populating the actual response node with information from the response;

adding the actual response node to the representation of the mark-up language document.

10. The method of claim 8 wherein the step of receiving information includes

receiving information from a response during run-time.

11. The method of claim 8 wherein the step of injecting the information at the location specified by the injection tag includes

injecting the received information into an internal representation of a mark-up document.

12. The method of claim 11 wherein the internal representation is one of a DOM representation, an XML tree, and other representations.

13. The method of claim 8 wherein the injected information is subsequently utilized during run-time as one of a part of a call and a part of a request.

14. The method of claim 13 wherein the request is a save request in UDDI.

15. A method for testing a Web server comprising the steps of:

a) generating a test suite file written in a markup language that includes at least one injection tag; and

b) at run-time receiving data that is unavailable when the test suite file is generated and modifying a representation of the test suite file with the received data.

16. The method of claim 15 further comprising the steps of:

c) using the received data in another portion of the test suite file.

17. A test infrastructure for interacting with a server that has capabilities comprising:

a) a test suite for use in testing the capabilities of the server; wherein the test suite includes an expected response for a first request and at least one reference to information not known to a tester when preparing the test suite; and

b) an injection module for receiving information from the server, for generating an actual response based on the received information, and for replacing the reference with a target in the actual response that is referenced by the reference.

18. The system of claim 17 further including:

c) a comparison module for comparing an actual response with an expected response.

19. The system of claim 17 further including:

a parser for receiving the test suite and generating a DOM representation of the test suite.

20. The system of claim 17 wherein the injection module further includes:

a node creator for creating a new node with the received information; and

a node adder for adding the newly created node to a representation of the test suite.